

DONALD R. ANTONELLI
DAVID T. TERRY
MELVIN KRAUS
WILLIAM I. SOLOMON*
GREGORY E. MONTONE
RONALD J. SHORE
DONALD E. STOUT
ALAN E. SCHIAVELLI
JAMES N. DRESSER
CARL I. BRUNDIDGE*
PAUL J. SKWIERAWSKI*

LAW OFFICES
ANTONELLI, TERRY, STOUT & KRAUS, LLP
SUITE 1800
1300 NORTH SEVENTEENTH STREET
ARLINGTON, VIRGINIA 22209

OF COUNSEL
CHITTARANJAN N. NIRMEL, PHD*
PATENT AGENT
LARRY N. ANAGNOS
TELEPHONE
(703) 312-6600
FACSIMILE
(703) 312-6666
E-MAIL
email@antonelli.com

RANDALL S. SVIHLA
DAVID S. LEE*
ROBERT M. BAUER
DEBORA J. MILLS
JENNIFER H. BUI*

December 21, 1998

Honorable Commissioner for Patents
Washington, D.C. 20231

Attorney Docket Number: 219.36435X00

Sir:

Attached please find the application papers of Jerrie L. COFFMAN and Brad R. RULLMAN, covering new and useful improvements in a EFFICIENTLY EXPORTING LOCAL DEVICE ACCESS ONTO A SYSTEM AREA NETWORK USING A DIRECT-CALL INTERFACE comprising:

Specification, (28) Claims, and Abstract
of the Disclosure (28 pages)

English language Declaration and Power of Attorney
(6 pages)

(6) Sheets of Drawings Showing Figures

Assignment and Recordation Form Cover Sheet

U.S. Government Filing Fee \$982.00

U.S. Government Recording Fee \$40.00

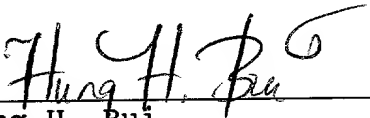
Change of Correspondence Address

12/21/98
89/512/60
OLD S. N. 0752C

Asst. Commissioner of Patents
Atty. Dkt. No.: 219.36435X00
December 21, 1998

Please charge any shortage in fees due in connection with the filing of this paper, to the Deposit Account of Antonelli, Terry, Stout & Kraus, LLP Account No. 01-2135 (219.36435X00) and please credit any overpayment of fees to such deposit account.

Respectfully submitted,



Hung H. Bul
Registration No. 40,415
ANTONELLI, TERRY, STOUT & KRAUS, LLP

HHB:dmw
Attachments

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Inventor(s): Jerrie L. COFFMAN and Brad R. RULLMAN

**Invention: EFFICIENTLY EXPORTING LOCAL DEVICE ACCESS ONTO A
SYSTEM AREA NETWORK USING A DIRECT-CALL
INTERFACE**

Antonelli, Terry, Stout & Kraus, LLP
Suite 1800
1300 North Seventeenth Street
Arlington, Virginia 22209
Tel: 703/312-6600
Fax: 703/312-6666

SPECIFICATION

Jerrie L. Coffman, a citizen of the United States, residing at 17795 NW Fall Court, Beaverton, Oregon 97006, and Brad R. Rullman, a citizen of the United States, residing at 1665 N.W. 131st Avenue, Portland, Oregon 97229, have invented certain new and useful improvements in the invention as titled above and of which the following is a specification.

EFFICIENTLY EXPORTING LOCAL DEVICE ACCESS ONTO A SYSTEM AREA NETWORK USING A DIRECT-CALL INTERFACE

Technical Field

The present invention relates to a direct-call interface between local and remote transports in a device driver for efficiently exporting local device access to a network.

Background

A computer network is a system designed to link together computers, servers, peripherals, storage devices, and communication devices for communications. Examples of such a computer network include a local area network (LAN), a wide are network (WAN), a campus area network (CAN), a metropolitan area network (MAN), and global area network (GAN). The central characteristics of a computer network is sharing, whether that be sharing of cost, resources, or data bases, while at the same time insuring a high degree of data privacy.

As high-speed and high-performance communications become necessary for many applications such as data warehousing, decision support, and transaction processing, many companies have adopted clustering technology for ability to provide availability and scalability for these applications. A cluster is a group of servers, workstations, and/or storage devices that are linked together to operate as a single system to deliver high performance, low latency, and extreme reliability. Clustering offers three primary benefits: scalability,

10
15
20

availability, and manageability. Scalability is obtained by allowing servers to work together and to allow additional services to be added for increased processing as needed. The cluster combines the processing power of all servers within the cluster to run a single logical application (such as a database server). Availability is obtained by allowing servers to "back each other up" in the case of failure. Likewise, manageability is obtained by allowing the cluster to be utilized as a single, unified computer resource, that is, the user sees the entire cluster (rather than any individual server) as the provider of services and applications.

High-performance network technologies known as system area networks (SANs) have recently been developed for linking servers and network-connected storage devices within a cluster. Virtual Interface (VI) Architecture is designed to enable applications to communicate over a system area network (SAN). Basically, the VI Architecture provides a transport service which allows a collection of independent standards-based servers to be utilized as a highly scalable cluster to meet the performance and capacity requirements of the largest and most demanding enterprise applications. Its fast server-to-server communications can enhance an application's scalability and performance in a variety of ways - from allowing a single application to run efficiently across dozens of clustered nodes, to speeding up the exchange of data between distributed application modules running on different application servers.

One of the inherent challenges of the system area network (SAN) is to design a data transport mechanism that can deliver a large amount of data between nodes in the cluster ("high bandwidth"), and that can exchange messages quickly between nodes in the cluster ("low latency"). Traditional data transports between nodes in a cluster are done through the network infrastructure provided by a host operating system (OS). The operating system (OS)

structure requires large amount of system processing overhead and extended processing time with respect to each message.

In particular, for a specific application such as access of a storage device of a remote server in a cluster, standard interfaces provided by the host operating system (OS) such as file system application program interfaces (API) are used to access a network file system (NFS). However, direct access to storage devices of remote servers within a cluster through input/output (I/O) subsystems was not performed transparently. Each request for a data transfer in the traditional data transports incurred a large amount of processing overhead of operating system (OS) stacks on both the local and remote servers of the cluster. This overhead limits input/output (I/O) bandwidth, increases input/output (I/O) latency, and increases the response time to the application.

Current alternative to high network overhead is for another application to generate special application-to-application messages to the remote node in order to access a remote server in a cluster. A remote application running on the remote node must issue an input/output (I/O) request to the remote operating system (OS) on behalf of the local application. This way the operating system (OS) overhead on the local server is avoided, but there are still a great deal of coordination between cooperating applications of the local server and the remote server.

Therefore, there is an urgent need for providing a direct, transparent access to storage devices connected to a host server within a network for efficient sharing of resources and databases among all network members.

SUMMARY

Accordingly, the present invention is designed to provide an exemplary input/output platform (IOP) access module in a host system for providing input/output device access between a host system and another system. An exemplary input/output platform (IOP) access module includes a Local Transport arranged to provide an interface to an input/output platform (IOP) supporting an array of input/output devices; a Remote Transport arranged to provide an interface to another system; and a Connection Manager arranged to establish connection services and to create a direct call path between the Local Transport and the Remote Transport so as to provide access to input/output devices.

BRIEF DESCRIPTION OF THE DRAWINGS

A more complete appreciation of exemplary embodiments of the present invention, and many of the attendant advantages of the present invention, will become readily apparent as the same becomes better understood by reference to the following detailed description when considered in conjunction with the accompanying drawings in which like reference symbols indicate the same or similar components, wherein:

FIG. 1 illustrates an exemplary background system area network (SAN) cluster of host and remote servers;

FIG. 2 illustrates an exemplary background server using a host operating system (OS) for providing network infrastructure for data transports between nodes in a cluster;

FIG. 3 illustrates an exemplary driver system for providing direct, transparent access to local storage devices connected to a host or remote server within a system area network

(SAN) cluster, for efficient sharing of resources and databases among all clustered servers according to the principles of the present invention;

FIG. 4 illustrates exemplary data structures established during initialization for a direct call interface between local and remote transports in an exemplary driver system according to the principles of the present invention;

FIG. 5 is a flowchart of an exemplary process of routing an inbound message through the local and remote transports in an exemplary driver system from a remote server within a system area network (SAN) according to the principles of the present invention; and

FIG. 6 is a flowchart of an exemplary process of routing an outbound message from a local I/O platform (IOP) of an exemplary driver system to a remote server within a system area network (SAN) according to the principles of the present invention.

DETAILED DESCRIPTION

While the following detailed description focuses on an exemplary implementation providing a connection arrangement between an input/output platform (IOP) of a host server and other remote servers of a system area network (SAN), the present invention is not limited thereto. More particularly, the present invention may be equally applicable for implementation with other types of networks or other types of device interconnections (e.g., non-networked and/or non-clustered devices), and between other types of devices.

Reference is now made to exemplary arrangements and particularly to FIG. 1 which illustrates an exemplary background cluster of a host server 10 and remote servers 14, 16, 18 connected to a system area network (SAN) 12. Each server is an autonomous system which can include one or more processors, memory, one or more input/output platforms and other

local I/O resources. Examples of such a server include desktop computers (PCs) and workstations. As shown in FIG.1, each server such as a host server 10 can include at least one SAN network interface card (NIC) 110 attached to a local bus, an input/output (I/O) platform (IOP) 120 of one or more input/output (I/O) processors, and an array of disk storage devices 130.

Further, each server, as shown in FIG. 2, can include one or more processors 112, a primary memory 114 containing application programs 115, a system manager 116, an operating system (OS) 117, and I/O driver modules 118 all interconnected to the I/O platform 120 of one or more I/O processors via a local bus 122. The operating system (OS) 117 provides traditional network infrastructure for data transports between nodes in a cluster using an industry standard Internet protocol (TCP/IP) and network file system (NFS) protocol stacks, although any other industry standard such as an Open Systems Interconnection (OSI) protocol stack may also be used for networking purposes. General descriptions of these software protocol stacks, particularly, the TCP/IP and NFS protocol stacks can be found in Tanenbaum, *"Computer Networks"* (Second Edition, Prentice Hall: 1988).

However, the exemplary background host operating system (OS) structure requires a large amount of system processing overhead and an extended processing time to process and send each message between the local and remote servers in a cluster. This overhead limits the I/O bandwidth and increases the I/O latency of the cluster. While there are alternatives to avoiding the operating system (OS) overhead on local and remote servers, such as coordinating special application-to-application messages between the local and remote servers, more effective solutions for providing direct, transparent access to storage devices connected

to servers within a system area network cluster are needed without incurring the overhead of the operating system (OS) protocol stack and without coordinating special application-to-application messages between nodes in a system area network (SAN) cluster.

Attention now turns to FIG. 3 which illustrates an exemplary driver system solution for providing a direct, transparent access to I/O storage devices connected to a host server 300 within a system area network (SAN) 330 cluster for efficient sharing of resources and databases among all clustered servers 300, 340, 350 according to the principles of the present invention. Each server as intended by the present invention is an autonomous system which may include one or more processors, a memory, one or more input/output platforms and other local I/O resources, such as a desktop computer (PC) or a workstation. The exemplary driver system according to the present invention is designed to avoid using special application-to-application messages between nodes within a SAN cluster and to bypass the layers of OS protocol stacks, thereby avoiding the overhead of the traditional network infrastructure in the operating system (OS) while concomitantly increasing the number of CPU cycles available for other data processes. The host server 300 as intended has a host operating system (OS) (not shown), and a host driver module 310 installed and dynamically configured to accept communication requests from remote servers 340 and 350 through a SAN network interface card (NIC) 328. Such a host driver module 310 (alternatively known as an input output platform (IOP) access module) may be provided as any one of a hardware module, a combined hardware/software module, and a software module. A software or partial software implementation is advantageous in that upgrades can be easily downloaded

(e.g., from a tangible medium such as a disk, or via the internet) for flexible/easy reconfiguration of the host driver module 310 and the host server 300.

As shown in FIG. 3, the exemplary driver system may include a host driver module 310 which resides on and which may interface with the host operating system (OS), a local bus 318 such as a Peripheral Component Interconnect (PCI) bus, and an Input/Output Platform (IOP) 320 for controlling an array of storage devices 326. Other popular bus varieties such as Extended Industry Standard Architecture (EISA), Video Electronics Standards Association (VESA) may be used for transporting data from the host device driver module 310 to the local IOP 320 and, vice versa. The disk storage devices 326 may include, for example, a magnetic storage device and/or an optical storage device for storing programs and databases. The Input/Output Platform (IOP) 320 may include at least one input/output (I/O) processor (not shown).

The IOP 320 contains a device driver module 322 which resides on and interfaces with the particular controller and storage devices, and a communication layer 324 which defines an open, standard mechanism for communication between the host driver module 310 and the device driver module 322. The device driver module 322 is responsible for control and data transfer of the hardware devices, such as storage devices. The communication layer 324 may include a message layer which sets up a communication session, and a transport layer which defines how information will be shared. Collectively, the communication layer 324 may be responsible for managing all requests, and providing a set of Application Programming Interfaces (APIs) for delivering messages, along with a set of support routines that process them.

The driver system, as shown in FIG. 3, may be designed in accordance with the Intelligent Input/Output (I₂O) Architecture Specification for allowing input/output (I/O) subsystems such as storage devices to operate independently from both the specific device being controlled and the host operating system (OS). Both the driver module 310 and the device driver module 322 of the IOP 320 may be provided as separate modules, or may be provided via a single device driver that is portable across a plurality of operating systems (OSs) and host network platforms, and may be designed to work interoperably with different combinations of I/O storage devices and operating systems (OSs). Typically, the host driver module 310 is the host OS-specific portion of the driver system that interfaces with the host OS, whereas the device driver module 322 is a device-specific portion of the driver system that interfaces with the particular controllers and I/O devices.

Again, the host driver module 310 as intended by an exemplary embodiment of the present invention is software including the following device driver components: a Connection Manager 312, a Local Transport 314, and a Remote Transport 316. However, hardware or mixed hardware/software equivalent devices may be optionally installed to perform the same functions of the host driver module 310 including the Connection Manager 312, the Local Transport 314, and the Remote Transport 316. In one exemplary embodiment of the present invention, the Local Transport 314 provides an interface to the IOP 320 on the PCI bus 318 and communicates with the IOP 320 across the PCI bus 318. The Remote Transport 316 provides an interface to other nodes such as remote servers 340 and 350 within the SAN cluster 330. The Connection Manager 312 provides connection services and the coordinating

function responsible for creating a direct call path between the Local Transport 314 and the Remote Transport 316.

The host server 300 and the exemplary driver system as shown in FIG. 3 may power-up and initialize independently with respect to other nodes such as remote servers 340 and 350 within the SAN cluster 330. The host server 300 is initialized by the specific OS software. When the exemplary driver system is initialized, the Local Transport 314 scans the PCI bus 318 to locate and initialize all local IOPs 320 (assuming more than one I/O processor is installed) and builds an opaque "context" structure for each IOP found, as will be later described in conjunction with FIG. 4. The Remote Transport 316 prepares to accept requests from a remote server through the SAN network interface card (NIC) 328. The Connection Manager 312 then queries the Local Transport 314 to determine the number of IOPs and builds a descriptor structure for each IOP. This IOP descriptor structure, as shown in FIG. 4, includes an exported table of function call pointers and the context required by the Local Transport 314 to communicate with each IOP. Next, the Connection Manager 312 establishes a SAN management communication channel through the Remote Transport 316, and waits for an external connection from a remote server on a SAN for exporting local device access onto the SAN 330 by way of a direct call interface between the Local Transport 314 and the Remote Transport 316.

Refer now to FIG. 4 which illustrates data structures established during initialization for a direct call interface between Local Transport 314 and Remote Transport 316 by the Connection Manager 312 for an inbound message from a remote SAN server for direct access to local IOP 320 in the exemplary driver system according to the present invention. The

Local Transport 314 builds a list of different IOP context structures if different IOPs are available in the host server 300. Each IOP context structure includes different designations for direct access to each IOPs. However, since only one IOP 320 is illustrated for purposes of discussion, a corresponding opaque IOP context structure is set for communication with the local IOP 320. In addition, the Local Transport 314 also has a send handler function which is a program interface to receive an inbound message from a remote SAN server for direct access to local IOP 320.

The Connection Manager 312 builds an IOP descriptor structure for each IOP found. Each IOP descriptor structure includes an exported table of function call pointers such as IOP context pointer and send handler function pointer required by the Local Transport 314 to communicate with the IOP 320. The Remote Transport 316 builds an IOP connection structure including at least an IOP descriptor pointer which refers to the IOP descriptor structure of the Connection Manager 312 for making a direct call to the Local Transport 314 through the send handler function. In addition, the Remote Transport 316 also has a receive handler function which is a program interface to receive an inbound message from a remote server on a SAN for direct access to local IOP 320 and to deliver an outbound message to a remote server on a SAN. For an outbound message to a remote server on a SAN, data structures established for a direct call interface between Local Transport 314 and Remote Transport 316 by the Connection Manager 312 as shown in FIG. 4 remain the same. However, the communication directions between the send handler function in the Local Transport 314 and the receive handler function in the Remote Transport 316 are reversed.

After initialization, the exemplary driver system as shown in FIG. 3 is prepared for providing a direct, transparent access to I/O storage devices within a system area network (SAN) cluster that bypasses the traditional OS protocol stacks for efficient sharing of resources so as to avoid incurring the overhead of the OS stack and coordinating special application-to-application messages between nodes of a SAN cluster. An exemplary direct call interface between the Local Transport 314 and the Remote Transport 316 of the host driver module 310 established for the purpose of exporting device access to host servers on a system area network (SAN) will be described in detail with reference to FIGs. 4, 5 and 6 hereinbelow.

First, in order to establish a service connection to an IOP 320, that is a logical connection of an IOP to a remote server for the purpose of sending messages and transporting data therebetween, a remote server 340 or 350 in a system area network (SAN) connects to and exchanges messages with the Connection Manager 312. Messages exchanged between the Connection Manager 312 of the exemplary driver system and the remote server are based on a protocol specified by the network used and/or the Intelligent I/O Architecture. The Connection Manager 312 next advertises the presence of local IOPs that are available for external use. In the exemplary embodiment of the present invention as shown in FIG. 3, only one IOP 320 is illustrated for purposes of discussion. Additional IOPs may be connected to PCI bus 318.

When a remote server 340 or 350 requests a service connection to an IOP 320, the Connection Manager 312 passes the address of an IOP descriptor structure as shown in FIG. 4 to the Remote Transport 316 to establish the service connection. The IOP descriptor

structure provides the Remote Transport 316 access to the function call pointers and context required by the Local Transport 314 to communicate directly with the IOP 320.

When a service connection to a local IOP 320 is established by remote server 340 or 350 and the Connection Manager 312 of the host server 300, low latency and high-bandwidth messages passing between nodes is obtained by bypassing the layers of OS protocol stacks when sending and receiving messages. Data structure pointers are exchanged to establish a direct call relationship between software modules on the host server 300 within a system area network (SAN) 330. To deliver an inbound message from a remote server on a system area network (SAN) 330, the receive handler function in the Remote Transport 314 simply refers to the IOP descriptor structure of the Connection Manager 312 to make a direct call to the send handler function in the Local Transport 314, providing the function with the IOP context and the message frame pointer. Likewise, an outbound message from the Local Transport 314 is delivered to a send handler function (not shown) in the Remote Transport 316. However, the outbound message includes a pointer to a structure containing the function address and context required by the Remote Transport 316 to send the message to a remote server 340 or 350 within a system area network (SAN) 330.

FIG. 5 illustrates an exemplary step-by-step process of routing an inbound message or data from a remote server 340 or 350 of a system area network (SAN) through the Local Transport 314 and Remote Transport 316 by the Connection Manager 312 of an exemplary driver system according to the principles of the present invention. Each message is transmitted in a series of frames. When a service connection is requested from a remote server, an inbound message frame from a remote server on the system area network (SAN) is

delivered to the receiver handler function in the Remote Transport 316 at step 510. The address of the IOP connection structure describing the service connection on which the inbound message was received (i.e., the context of the service connection) is located at the Remote Transport 316 at step 520. The IOP descriptor structure is located by using an IOP
5 descriptor pointer to refer to the IOP descriptor structure of the Connection Manager 312.

Next, a context field in the inbound message frame is saved and replaced with a new context field for the Remote Transport 316 at step 530. The pointer to the IOP descriptor structure of the Connection Manager 312 is retrieved from the IOP connection structure at step 540. Then, the send handler function pointer and the IOP context pointer are read from the IOP descriptor structure of the Connection Manager 312 at step 550. The send handler
10 function in the Local Transport 314 is called in order to pass the inbound message frame using the IOP context pointer in step 560.

Likewise, FIG. 6 illustrates an exemplary step-by-step process of routing an outbound message or data from a local I/O platform (IOP) 320 of an exemplary driver system to a remote server of a system area network (SAN) according to the principles of the present invention. Again, each message is transmitted in a series of message frames. When an
15 outbound message is routed to a remote server on a system area network (SAN), an outbound message frame is delivered to the send handler function in the Local Transport 314 at step 610. The context field in the outbound message frame is read at step 620. This context is a
20 pointer to a callback object structure in the Remote Transport 316 which contains the address of the Remote Transport IOP connection structure for a service connection.

Next, the callback function is called, passing the callback context pointer and the outbound message frame as outgoing parameters at step 630. At the Remote Transport 316, an outgoing message frame and the IOP connection structure address are delivered to the send handler function at step 640. A context field in the outgoing message frame is replaced with the saved context from the original inbound message frame as described with reference to FIG. 5, at step 650. Then, the outgoing message frame is sent out the service connection to a remote server on a SAN at step 660.

As described, the present invention provides a sophisticated driver system of a server system that is designed to bypass the layers of OS protocol stacks without incurring the overhead of the traditional network infrastructure in the operating system (OS), and to avoid using special application-to-application messages between nodes within a SAN cluster. The driver system may be configured in accordance with an Intelligent Input/Output (I₂O) Architecture specification with a host driver module and a device driver module interconnected by a communication layer for available operations with different combinations of I/O storage devices and operating systems (OSs).

While there have been illustrated and described what are considered to be exemplary embodiments of the present invention, it will be understood by those skilled in the art and as technology develops that various changes and modifications may be made, and equivalents may be substituted for elements thereof without departing from the true scope of the present invention. In addition, many modifications may be made to adapt a particular situation to the teachings of the present invention without departing from the central scope thereof.

Therefore, it is intended that the present invention not be limited to any particular exemplary

embodiment disclosed, but that the present invention includes all embodiments falling within the scope of the appended claims.

What is claimed is:

CLAIMS

1 1. An input/output platform (IOP) access module for providing input/output device
2 access between a host system and another system comprising:
3 a Local Transport arranged to provide an interface to an input/output platform (IOP)
4 supporting an array of input/output devices;
5 a Remote Transport arranged to provide an interface to said another system; and
6 a Connection Manager arranged to establish connection services and to create a direct
7 call path between the Local Transport and the Remote Transport so as to provide access to
8 input/output devices.

1 2. The input/output platform (IOP) access module of claim 1, wherein said IOP
2 access module is one of a hardware module, a combined hardware/software module, and a
3 software module provided on a tangible medium.

1 3. The input/output platform (IOP) access module of claim 1, wherein said host
2 system corresponds to a host server, said another system corresponds to any one of remote
3 servers, with said host server and said remote servers being arranged on a computer network.

1 4. The input/output platform (IOP) access module of claim 2, further comprising said
2 IOP which comprises:
3 at least one or more input/output processors;

4 at least one storage device as said input/output devices;
5 a device driver module arranged to interface with said storage device;
6 a communication layer which defines a mechanism for communications between the
7 Local Transport and the device driver module.

1 5. The input/output platform (IOP) access module of claim 4, wherein said
2 communication layer is responsible for managing all service requests and providing a set of
3 Application Programming Interfaces (APIs) for delivering messages, along with a set of
4 support routines that process the messages.

2 6. The input/output platform (IOP) access module of claim 5, wherein said
3 communication layer comprises a message layer which sets up a communication session, and
4 a transport layer which defines how information will be shared.

2 7. A host system, comprising:
3 a processor;
4 an array of storage devices;
5 a driver module for exporting local storage device access onto a computer network,
6 said driver module comprising:
7 a device driver module arranged to provide an interface to said array of local
storage devices;

1 a host driver module arranged to provide an interface to an operating system,
2 said system driver module comprising a Local Transport which communicates with the
3 device driver module, a Remote Transport which provides an interface to said
4 computer network, and a Connection Manager which establishes connection services
5 with remote systems on said computer network and which creates a direct call path
6 between the Local Transport and the Remote Transport to provide access to said
7 storage devices; and

8 a communication layer which supports communications between the host driver
9 module and the device driver module.

10 8. The host system of claim 7, wherein said host system corresponds to a host server,
11 said remote systems corresponds to remote servers arranged in a cluster, and said computer
12 network corresponds to a system area network for communications between said host system
13 and said remote systems within said cluster.

14 9. The host system of claim 7, wherein said communication layer is responsible for
15 managing all service requests and providing a set of Application Programming Interfaces
16 (APIs) for delivering messages, along with a set of support routines that process the
17 messages.

1 10. The host system of claim 9, wherein said communication layer comprises a
2 message layer which sets up a communication session, and a transport layer which defines
3 how information will be shared.

1 11. The host system of claim 9, wherein said system driver module and said device
2 driver module constitute a single device that is portable across a plurality of operating
3 systems and host network platforms, and works interoperably with a plurality of storage
4 devices and operating systems.

1 12. The host system of claim 9, wherein said system driver module and said device
2 driver module operate in accordance with an Intelligent Input/Output (I₂O) specification for
3 allowing storage devices to operate independently from the operating system.

1 13. The host system of claim 7, wherein said driver module is one of a hardware
2 module, a combined hardware/software module, and a software module provided on a
3 tangible medium.

1 14. A driver configuration of a host server for exporting local storage device access
2 onto a computer network, comprising:
3 an input/output platform (IOP) arranged to control an array of local storage devices;
4 and
5 a system driver module comprising:

6 a Local Transport arranged to provide an interface to said input/output
7 platform (IOP);

8 a Remote Transport arranged to provide an interface to said computer
9 network; and

10 a Connection Manager arranged to establish connection services with
11 remote servers on said computer network and coordinate functions responsible
12 for creating a direct call path between the Local Transport and the Remote
13 Transport to provide access to the local storage devices.

14 15. The driver configuration of claim 14, wherein said input/output platform (IOP)
15 supports at least one or more input/output processors, and comprises:

16 a device driver module which interfaces the local storage devices for controlling said
17 array of local storage devices; and

18 a communication layer which defines a mechanism for communications between the
19 system driver module and the device driver module.

20 16. The driver configuration of claim 15, wherein said communication layer is
21 responsible for managing and dispatching all service requests and providing a set of
22 Application Programming Interfaces (APIs) for delivering messages, along with a set of
23 support routines that process the messages, and is comprised of a message layer which sets up
24 a communication session, and a transport layer which defines how information will be shared.

1 17. The driver configuration of claim 15, wherein said system driver module and said
2 device driver module constitute a single device that is portable across a plurality of host
3 operating systems and host network platforms, and works interoperably with a plurality of
4 storage devices and host operating systems.

1 18. The driver configuration of claim 15, wherein said system driver module and said
2 device driver module operate in accordance with an Intelligent Input/Output (I₂O)
3 specification for allowing storage devices to operate independently from the operating system
4 of the host server.

1 19. The driver configuration of claim 15, wherein, upon initialization, said Local
2 Transport scans the local bus so as to locate and initialize all local input/output platforms
3 (IOPs) and builds an opaque "context" structure for each input/output platform (IOP),
4 wherein said Remote Transport prepares to accept requests from a remote server through said
5 computer network, and wherein said Connection Manager queries said Local Transport so as
6 to determine the number of input/output platforms (IOPs), builds an IOP descriptor structure
7 for each input/output platform (IOP) which includes an exported table of function call
8 pointers and the context required by the Local Transport to communicate with the
9 input/output platform (IOP), and finally establishes a network management communication
10 channel through the Remote Transport, which waits for an external connection from said
11 remote server on said computer network for exporting local device access onto said computer
12 network using said direct call path between the Local Transport and the Remote Transport.

1 20. The driver configuration of claim 19, wherein said Local Transport further has a
2 send handler function and said Remote Transport further has a receive handler function which
3 are respective program interfaces for receiving an inbound message from a remote server on
4 said computer network for direct access to local input/output platform and for delivering an
5 outbound message to said remote server on said computer network.

1 21. The driver configuration of claim 19, wherein said Remote Transport further
2 builds an IOP connection structure including at least an IOP descriptor pointer which refers to
3 the IOP descriptor structure of the Connection Manager for making a direct call to the Local
4 Transport through the receive handler function and the send handler function.

1 22. A process of exporting storage device access onto a computer network using an
2 input/output platform (IOP) access module of a host server, comprising the steps of:

3 providing an interface to an input/output platform (IOP) supporting an array of storage
4 devices;

5 providing an interface to a remote server on said computer network;

6 establishing service connection between said host server and said remote server on
7 said computer network in response to a request from a remote server on said computer
8 network; and

9 providing a direct call access to said storage devices for said remote server to share
10 resources of said storage devices while bypassing operating system protocol stacks.

1 23. A process of establishing a service connection to a local input/output platform
2 (IOP) connected to a local bus using a driver module in response to a request from a remote
3 server on a system area network, comprising the steps of:

4 beginning initialization of said driver module which provides access to a local storage
5 system while bypassing protocol stacks of a host operating system, said driver module
6 comprising a Local Transport which provides direct access to the local storage device system,
7 a Remote Transport which interfaces to other nodes of said system area network, and a
8 Connection Manager which provides connection services and coordinating functions
9 responsible for creating a direct call path between the Local Transport and the Remote
10 Transport;

11 scanning, at said Local Transport, the local bus to locate and initialize all local
12 input/output platforms (IOPs), and building an IOP context structure for each input/output
13 platform (IOP) found;

14 preparing, at said Remote Transport, to accept a request for a service connection from
15 said remote server on said system area network;

16 asking, at said Connection Manager, whether said Local Transport determines the
17 number of input/output platforms (IOPs), and building a descriptor structure for each
18 input/output platform (IOP) which includes an exported table of function call pointers and the
19 context required by the Local Transport to communicate with the input/output platform (IOP);
20 and

21 establishing a system area network management communication channel through the
22 Remote Transport, which waits for an external connection from said remote server on said

23 system area network for exporting local device access onto said system area network using
24 said direct call path between the Local Transport and the Remote Transport.

1 24. The process of claim 23, wherein said input/output platform (IOP) comprises:
2 a device driver module which interfaces the local storage devices, and which controls
3 an array of local storage devices; and
4 a communication layer which defines a mechanism for communication between the
5 system driver module and the device driver module.

1 25. The process of claim 24, wherein said communication layer is responsible for
2 managing and dispatching all service requests and providing a set of Application
3 Programming Interfaces (APIs) for delivering messages, along with a set of support routines
4 that process the messages, and is comprised of a message layer which sets up a
5 communication session, and a transport layer which defines how information will be shared.

1 26. The process of claim 23, wherein said system driver module and said device
2 driver module constitute a single device that is portable across a plurality of host operating
3 systems and host network platforms, and operate in accordance with an Intelligent
4 Input/Output (I₂O) specification for allowing storage devices to operate independently from
5 the host operating system.

1 27. The process of claim 23, wherein said Local Transport further has a send handler
2 function and said Remote Transport further has a receive handler function which are
3 respective program interfaces for receiving an inbound message from a remote server on said
4 system area network for direct access to local input/output platform (IOP) and for delivering
5 an outbound message to said remote server on said system area network.

1 28. The process of claim 27, wherein said Remote Transport further builds an IOP
2 connection structure including at least an IOP descriptor pointer which refers to the IOP
3 descriptor structure of the Connection Manager for making a direct call to the Local
4 Transport through the receive handler function and the send handler function.

ABSTRACT OF THE DISCLOSURE

A novel module system solution is provided for direct, transparent access to I/O storage devices connected to a host server within a system area network cluster for efficient sharing of resources and databases among all clustered servers. An exemplary driver system comprises a host driver module which may reside on and which may interface to a host operating system, and which establishes service connections with remote data processors on the system area network and provides direct access to the local storage devices while bypassing protocol stacks of the host operating system; an input/output platform (IOP) including a device driver module which may reside on and which may interface the local storage devices for controlling an array of local storage devices; and a local bus which connects and transports messages and data between the host driver module and the input/output platform (IOP). The host driver module may be a software stack which includes a Local Transport for providing an interface to the input/output platform (IOP) on the local bus and communicating with the input/output platform (IOP) across the local bus; a Remote Transport for providing an interface to other nodes of the system area network; and a Connection Manager for providing connection services and coordinating functions responsible for creating a direct call path between the Local Transport and the Remote Transport. The input/output platform (IOP) includes a communication layer which defines an open, standard mechanism for communication between the host driver module and the device driver module. Both the host driver module and the device driver module may constitute a single device that is portable across a plurality of operating systems (OSs) and host network platforms.

FIG. 1

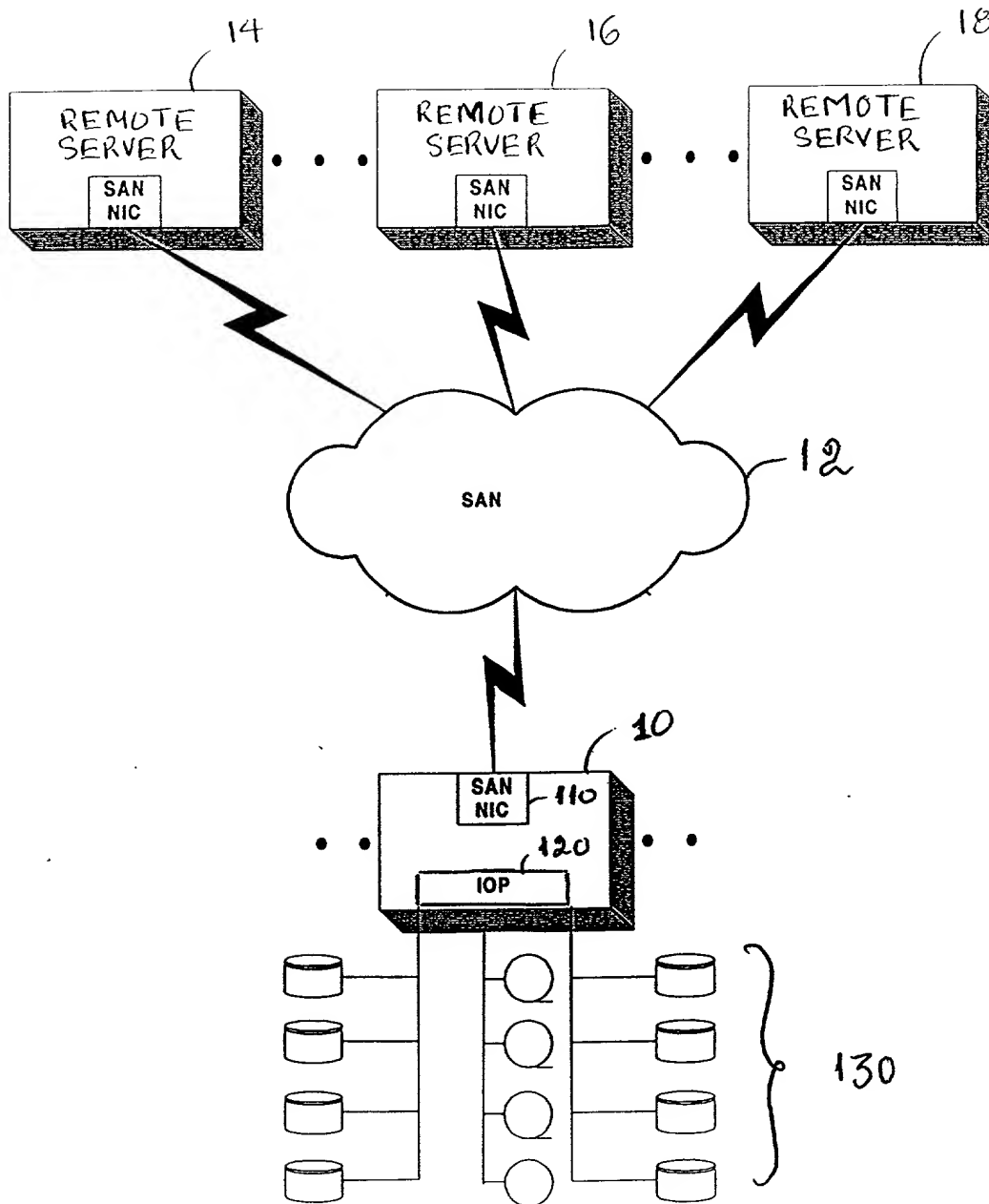


FIG. 2

10
3

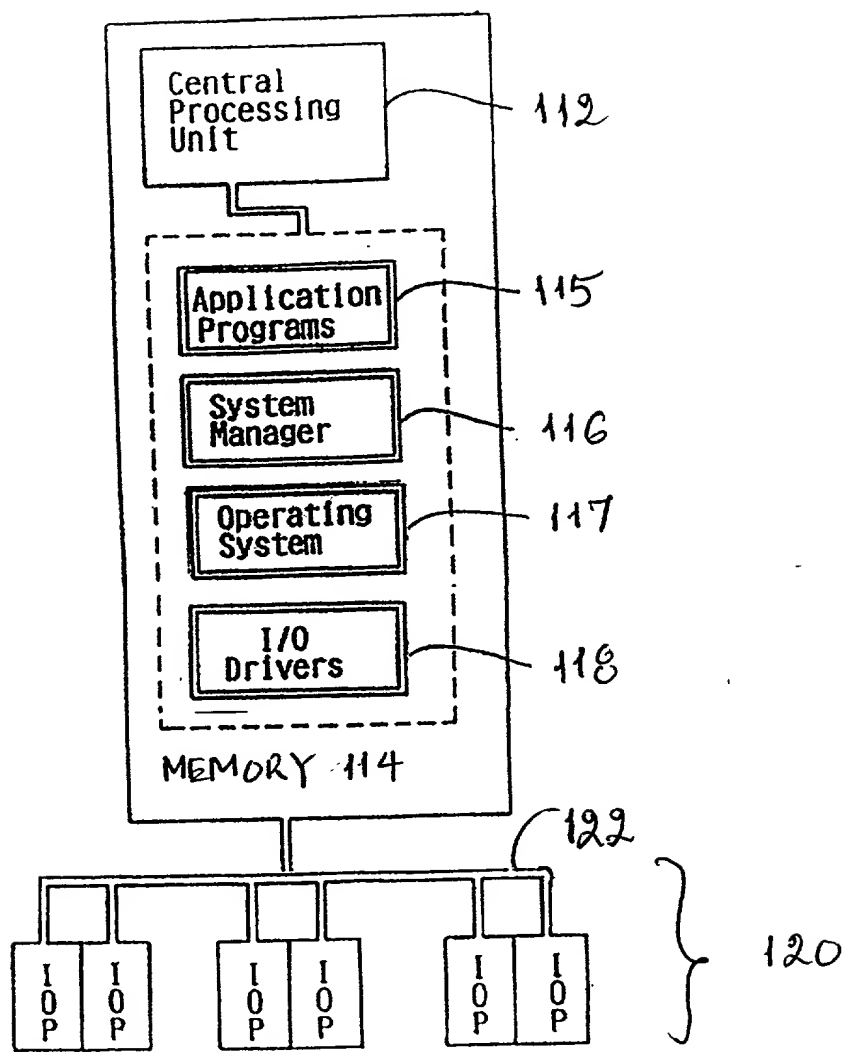


FIG. 3

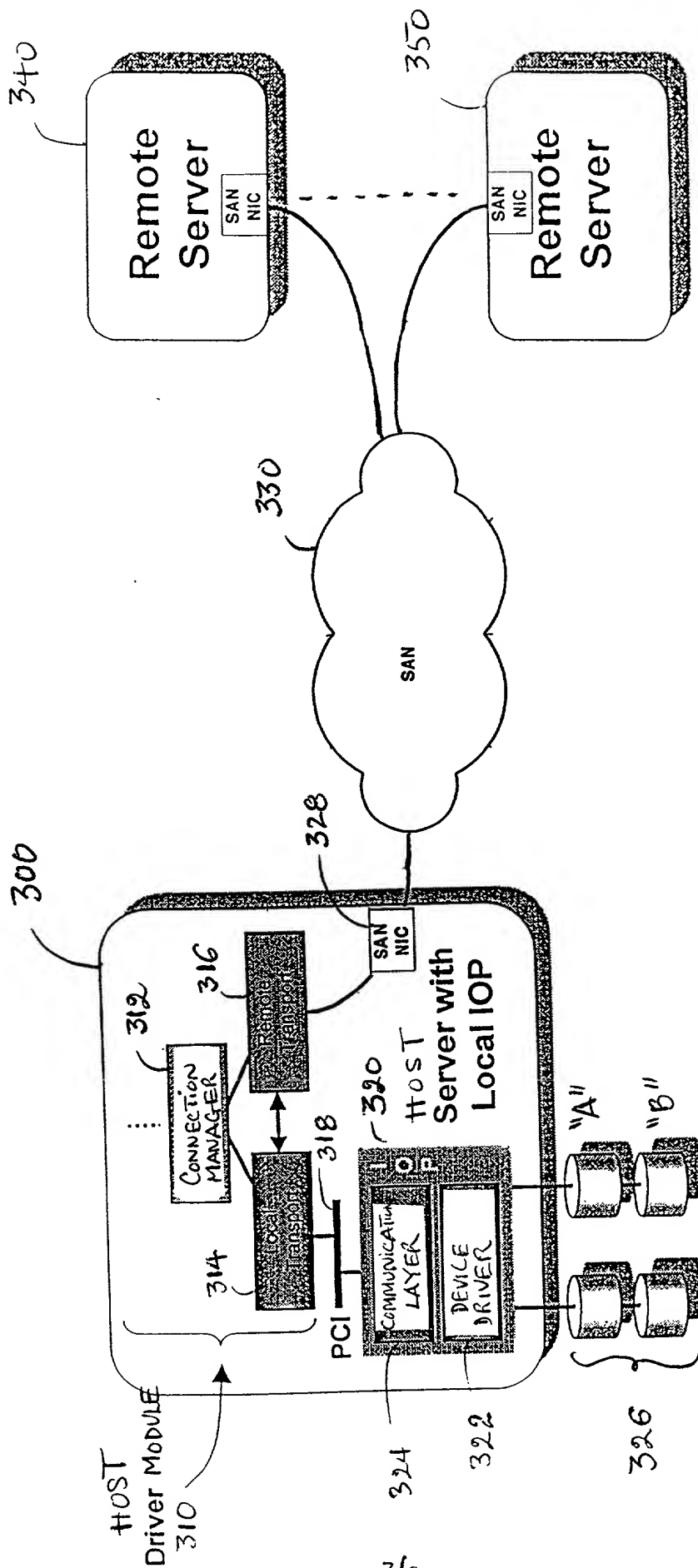


FIG. 4

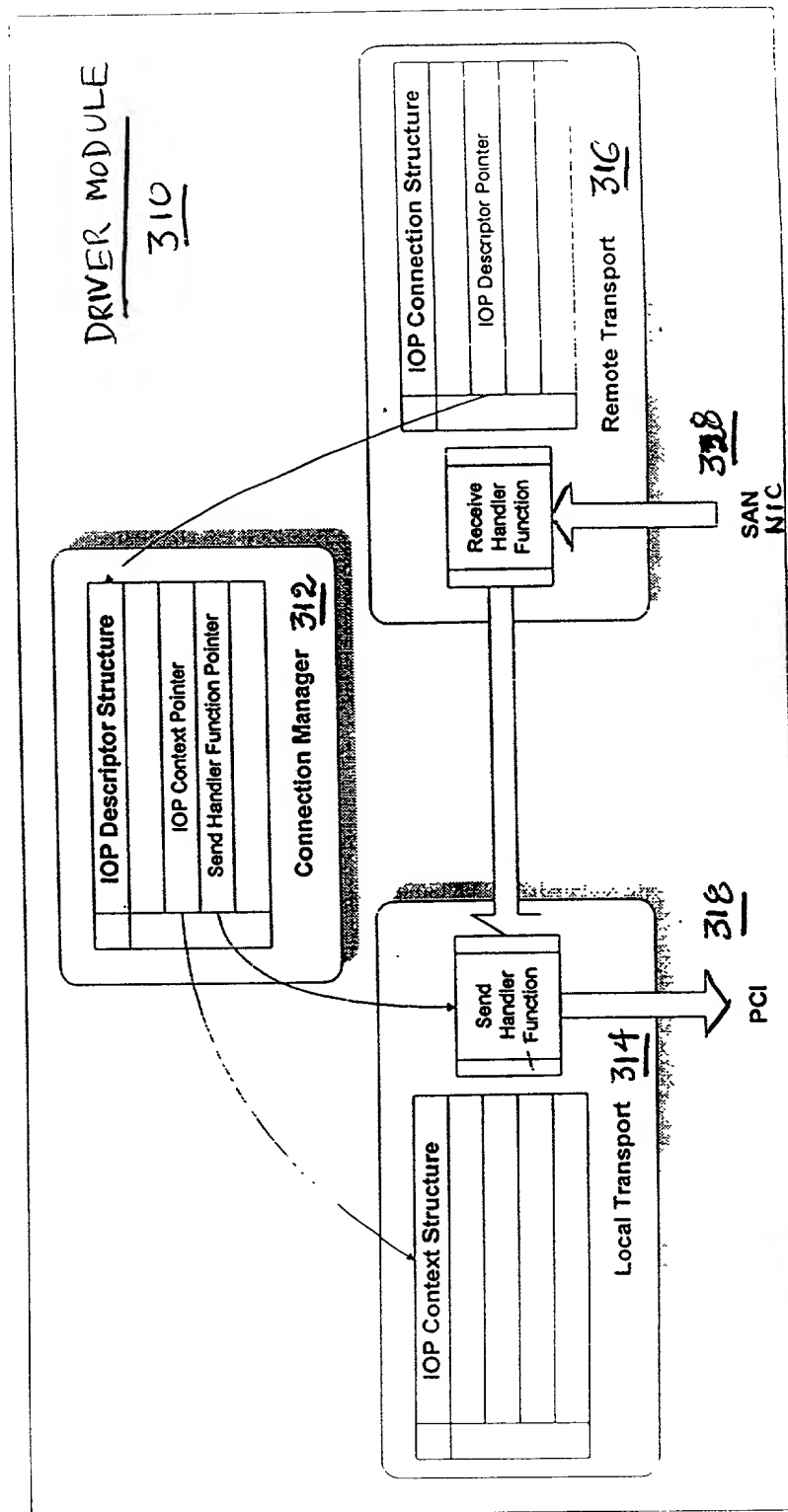
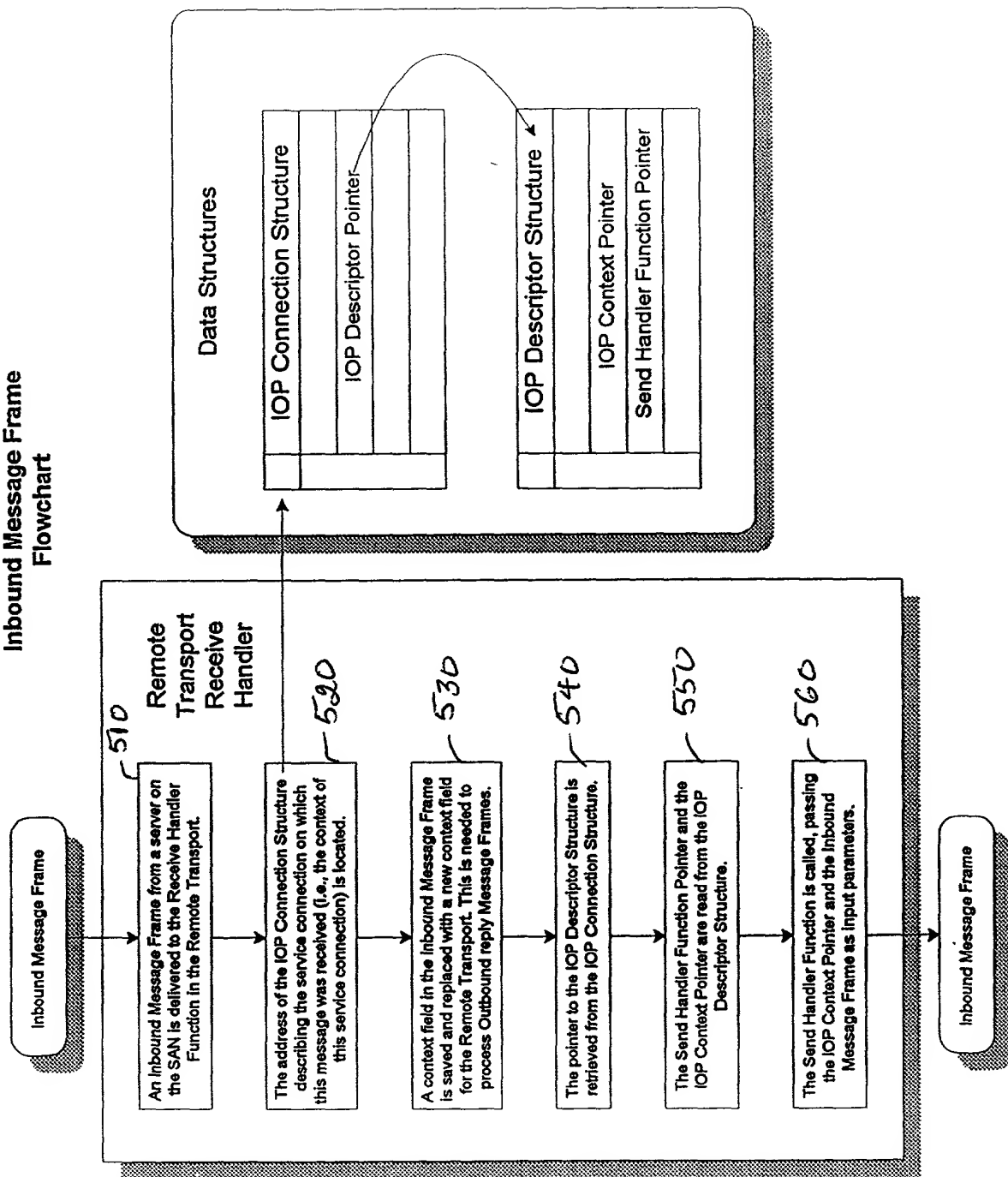
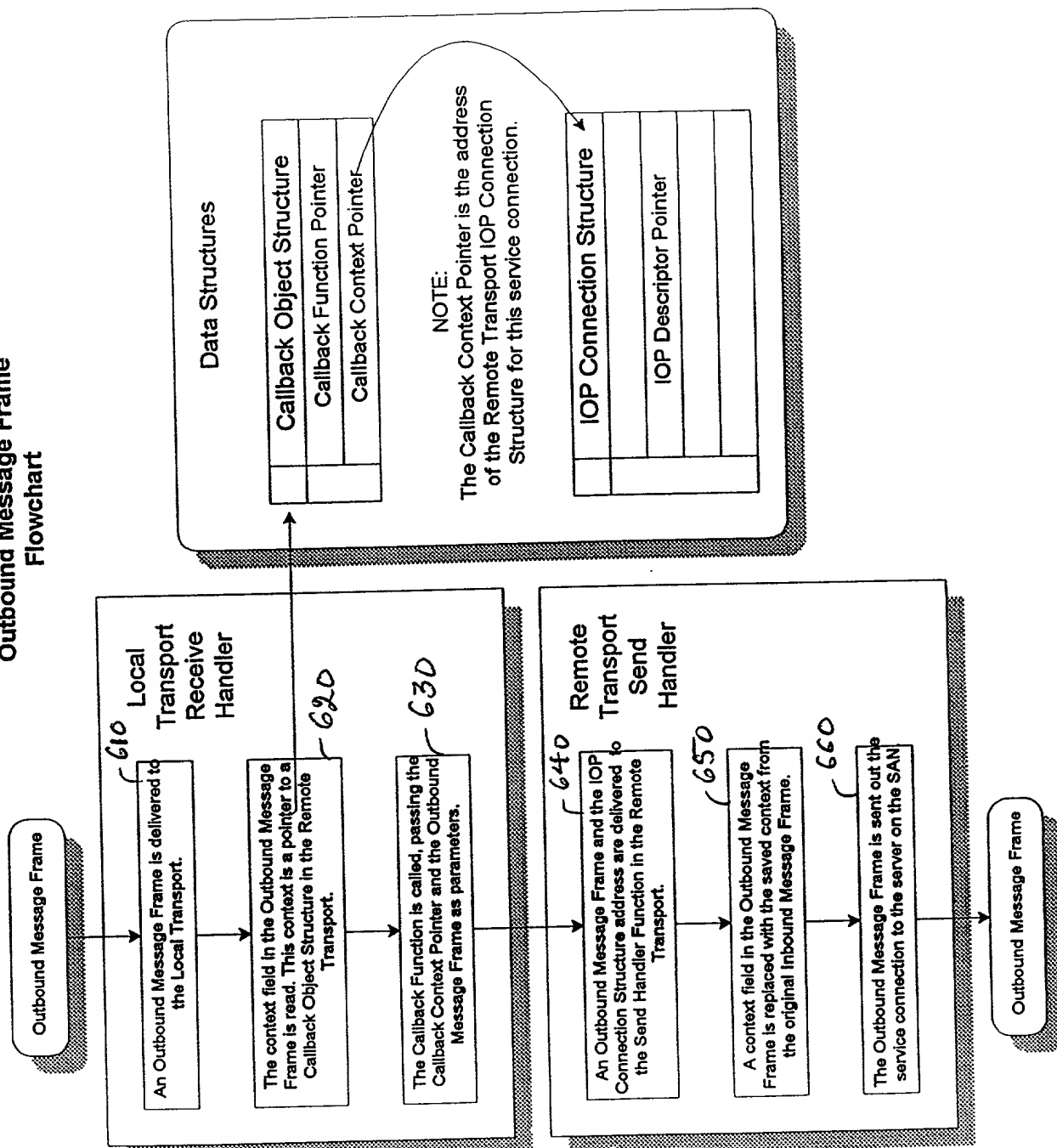


FIG. 5

Inbound Message Frame Flowchart



Outbound Message Frame Flowchart



PATENT

DECLARATION AND POWER OF ATTORNEY FOR PATENT APPLICATION
(FOR INTEL CORPORATION PATENT APPLICATIONS)

I believe I am the original, first, and sole inventor (if only one name is listed below) or an original, first, and joint inventor (if plural names are listed below) of the subject matter which is claimed and for which a patent is sought on the invention entitled EFFICIENTLY EXPORTING LOCAL DEVICE ACCESS ONTO A SYSTEM AREA NETWORK USING A DIRECT-CALL INTERFACE, the specification of which

I hereby claim foreign priority benefits under Title 35, United States Code, Section 119(a)-(d), of any foreign application(s) for patent or inventor's certificate listed below and have also identified below any foreign application for patent or inventor's certificate having a filing date before that of the application on which priority is claimed:

Prior Foreign Application(s)

Priority
Claimed

_____ (Number)	_____ (Country)	_____ (Day/Month/Year Filed)	Yes	No
_____ (Number)	_____ (Country)	_____ (Day/Month/Year Filed)	Yes	No
_____ (Number)	_____ (Country)	_____ (Day/Month/Year Filed)	Yes	No

I hereby claim the benefit under title 35, United States Code, Section 119(e) of any United States provisional application(s) listed below

_____ (Application Number)	_____ Filing Date
_____ (Application Number)	_____ Filing Date

I hereby claim the benefit under Title 35, United States Code, Section 120 of any United States application(s) listed below and, insofar as the subject matter of each of the claims of this application is not disclosed in the prior United States application in the manner provided by the first paragraph of Title 35, United States Code, Section 112, I acknowledge the duty to disclose all information known to me to be material to patentability as defined in Title 37, Code of Federal Regulations, Section 1.56 which became available between the filing date of the prior application and the national or PCT international filing date of this application:

_____ (Application Number)	_____ Filing Date	_____ (Status – patented, pending, abandoned)
_____ (Application Number)	_____ Filing Date	_____ (Status – patented, pending, abandoned)

I hereby appoint Donald R. Antonelli, Reg. No. 20,296; David T. Terry, Reg. No. 20,178; Melvin Kraus, Reg. No. 22,266; William I. Solomon, Reg. No. 28,565; Gregory E. Montone, Reg. No. 28,141; Ronald J. Shore, Reg. No. 28,577; Donald E. Stout, Reg. No. 26,422; Alan E. Schiavelli, Reg. No. 32,087; James N. Dresser, Reg. No. 22,973; Carl I. Brundidge, Reg. No. 29,621 and Paul J. Skwierawski, Reg. No. 32,173 as my patent attorneys/agents of Antonelli, Terry, Stout & Kraus, LLP, with offices located at 1300 N. 17th Street, Suite 1800, Arlington, VA 22209, and Alan K. Aldous, Reg. No. 31,905; Robert D. Anderson, Reg. No. 33,826; Joseph R. Bond, Reg. No. 36,458; Richard C. Calderwood, Reg. No. 35,468; Cynthia Thomas Faatz, Reg. No. 39,973; Sean Fitzgerald, Reg. No. 32,027; Seth Z. Kalson, Reg. No. 40,670; David J. Kaplan, Reg. No. 41,105; Leo V. Novakoski, Reg. No. 37,198; Naomi Obinata, Reg. No. 39,320; Thomas C. Reynolds, Reg. No. 32,488; Steven P. Skabrat, Reg. No. 36,279; Howard A. Skaist, Reg. No. 36,008; Steven C. Stewart, Reg. No. 33,555; Raymond J. Werner, Reg. No. 34,752; and Charles K. Young, Reg. No. 39,435; my patent attorneys, and Jeffrey S. Draeger, Reg. No. 41,000; Thomas Raleigh Lane, Reg. No. P42,781; Calvin E. Wells, Reg. No. P43,256; and Alexander Ulysses Witkowski, Reg. No. P43,280; my patent agents, of INTEL CORPORATION; with full power of substitution and revocation, to prosecute this application and to transact all business in the Patent and Trademark Office connected herewith.

Send correspondence to _____
(Name of Attorney or Agent)

ANTONELLI, TERRY, STOUT & KRAUS, LLP, 1300 N. 17th Street, Suite 1800, Arlington, Virginia 22209 and direct telephone calls to Hung H. BUI, tel: 703/312-6600, fax: 703/312-6666.
(Name of Attorney or Agent)

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

Full Name of Sole/First Inventor Jerrie L. COFFMAN
Inventor's Signature Jerrie L. Coffman Date 12/17/98
Residence Beaverton, Oregon Citizenship United States
(City, State) (Country)
Post Office Address 17795 NW Fall Court, Beaverton, Oregon 97006, USA

Full Name of Second/Joint Inventor Brad R. RULLMAN
Inventor's Signature Brad R. Rullman Date 12/17/98
Residence Portland, Oregon Citizenship United States
(City, State) (Country)
Post Office Address 1665 N.W. 131st Avenue, Portland, Oregon 97229, USA

Full Name of Third/Joint Inventor _____
Inventor's Signature _____ Date _____
Residence _____ Citizenship _____
(City, State) (Country)

Post Office Address _____

Full Name of Fourth/Joint Inventor _____

Inventor's Signature _____ Date _____

Residence _____ (City, State) _____ Citizenship _____ (Country)

Post Office Address _____

INTEL CORPORATION
Rev. 08/05/98 (D3 INTEL)

Full Name of Fifth/Joint Inventor _____

Inventor's Signature _____ Date _____

Residence _____ (City, State) _____ Citizenship _____ (Country) _____

Post Office Address _____

Full Name of Sixth/Joint Inventor _____

Inventor's Signature _____ Date _____

Residence _____ (City, State) _____ Citizenship _____ (Country) _____

Post Office Address _____

Full Name of Seventh/Joint Inventor _____

Inventor's Signature _____ Date _____

Residence _____ (City, State) _____ Citizenship _____ (Country) _____

Post Office Address _____

Title 37, Code of Federal Regulations, Section 1.56
Duty to Disclose Information Material to Patentability

(a) A patent by its very nature is affected with a public interest. The public interest is best served, and the most effective patent examination occurs when, at the time an application is being examined, the Office is aware of and evaluates the teachings of all information material to patentability. Each individual associated with the filing and prosecution of a patent application has a duty of candor and good faith in dealing with the Office, which includes a duty to disclose to the Office all information known to that individual to be material to patentability as defined in this section. The duty to disclose information exists with respect to each pending claim until the claim is cancelled or withdrawn from consideration, or the application becomes abandoned. Information material to the patentability of a claim that is cancelled or withdrawn from consideration need not be submitted if the information is not material to the patentability of any claim remaining under consideration in the application. There is no duty to submit information which is not material to the patentability of any existing claim. The duty to disclose all information known to be material to patentability is deemed to be satisfied if all information known to be material to patentability of any claim issued in a patent was cited by the Office or submitted to the Office in the manner prescribed by §§1.97(b)-(d) and 1.98. However, no patent will be granted on an application in connection with which fraud on the Office was practiced or attempted or the duty of disclosure was violated through bad faith or intentional misconduct. The Office encourages applicants to carefully examine:

- (1) Prior art cited in search reports of a foreign patent office in a counterpart application, and
- (2) The closest information over which individuals associated with the filing or prosecution of a patent application believe any pending claim patentably defines, to make sure that any material information contained therein is disclosed to the Office.

(b) Under this section, information is material to patentability when it is not cumulative to information already of record or being made of record in the application, and

- (1) It establishes, by itself or in combination with other information, a prima facie case of unpatentability of a claim; or
- (2) It refutes, or is inconsistent with, a position the applicant takes in:
 - (i) Opposing an argument of unpatentability relied on by the Office, or
 - (ii) Asserting an argument of patentability.

A prima facie case of unpatentability is established when the information compels a conclusion that a claim is unpatentable under the preponderance of evidence, burden-of-proof standard, giving each term in the claim its broadest reasonable construction consistent with the specification, and before any consideration is given to evidence which may be submitted in an attempt to establish a contrary conclusion of patentability.

(c) Individuals associated with the filing or prosecution of a patent application within the meaning of this section are:

- (1) Each inventor named in the application;
 - (2) Each attorney or agent who prepares or prosecutes the application; and
 - (3) Every other person who is substantively involved in the preparation or prosecution of the application and who is associated with the inventor, with the assignee or with anyone to whom there is an obligation to assign the application.
- (d) Individuals other than the attorney, agent or inventor may comply with this section by disclosing information to the attorney, agent, or inventor.